

WEB-программирование семинар 5

Аржанников Дмитрий

reddysponkey@gmail.com

+7 915 306-94-72

Ильин Илья

skamper2000@mail.ru

+7 965 432-44-97

ИДМ-19-01

(магистратура)



Содержание

| Основные понятия

| Подключение сценариев к html-документу

| Типы данных и переменные в JavaScript

| Выражения в JavaScript

| Циклы JavaScript

| Манипуляции с DOM



Основные понятия

PHP

PHP (изначально *Personal Home Page Tools* – “Инструменты для создания персональных веб-страниц”) – скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов.



Как записывается PHP

PHP скрипт начинается с `<?php` и заканчивается на `?>`. Все, что между `<?php` и `?>`, это PHP код.

Файл, в котором записан PHP код нужно сохранять с расширением `.php`.



Вывод текста на экран. Оператор echo.

Наиболее употребляемым оператором для вывода текста на экран является оператор **echo**.

```
<?php  
echo "Привет от PHP";  
echo 2019;  
echo 'Петров Иван<br>Родился<br>...';  
?>
```



Синтаксис HEREDOC

Для отображения большого количества текста используют т.н. синтаксис **heredoc**. Он начинается с символов **<<<**, после которых может быть записан произвольный идентификатор. После располагаемого текста стоит указать тот самый идентификатор, что и в начале кода.

```
<?php
echo <<<END
<p>Для отображения большого <br> количества текста используют
синтаксис heredoc</p>
END;
?>
```



Комментарии

В PHP существует 3 типа комментариев.
Вложенные комментарии недопустимы.

```
<?php
/* 1.Многострочный
комментарий */

// 2.Однострочный комментарий

# 3. Тоже однострочный комментарий
?>
```



Переменные в РНР

Переменные в PHP

Синтаксис переменной состоит из знака доллара – \$ и "свободного" идентификатора которому присваивается какое-нибудь значение.

Имя (идентификатор) переменной чувствительно к регистру и не может начинаться с цифр и пробела.

```
<?php  
$name = "Виктор";  
?>
```



Создание переменной

Переменная создается тогда, когда ей присваивают какое-нибудь значение.

Для присвоения значения переменной используют оператор присвоения `=`.

```
<?php
$surname = "Петров";
$number = 1269794645;
$pi = 3.14159265;
$hello = "Hi all";
?>
```



Вывод переменной

Переменную можно вывести с помощью оператора `echo`.

```
<?php
$name = "Виктор";

echo "Ваше имя ", $name, "<br>";
?>
```

```
<?php
$bann = 5; // Бананы
$lem = 10; // Лимоны
$together = $bann + $lem; // Всего

echo "Количество фруктов ", $together;
?>
```



Интерполяция переменных

```
<?php
$capital = "Paris";
// Значение переменной может быть отображено например так:
echo "The capital of France is ", $capital,"<br />";
// Но есть способ сделать это проще. Если имя переменной заключено
// в двойные (не одинарные) кавычки, то переменная интерполируется. Например:
echo "The capital of France is $capital <br />";
?>
```

Также существует одна тонкость при использовании интерполяции переменных. Например:

```
<?php
$text = "news";
echo "Where's the $textpaper <br />";
?>
```

```
<?php
$text = "news";
echo "Where's the {$text}paper <br />";
?>
```



Переменные, содержащие имена других переменных

```
<?php
$apples = 5;      // Создаем переменную $apples
$fruit = "apples"; /* Создаем переменную $fruit, которая содержит имя переменной $apples */

// Сейчас мы можем вывести $apples, как $$fruit

echo "Число яблок - ", $$fruit;
?>
```

```
<?php
echo "Число яблок - ${$fruit}";
?>
```



Константы в PHP

```
<?php  
define("pi", 3.14);  
?>
```

```
<?php  
define("pi", 3.14);  
  
echo "Математическая константа Пи равняется ", pi;  
?>
```



Типы данных в PHP

◆ Boolean

◆ Integer

◆ String

◆ Float

◆ Object

◆ Array

◆ Resource

◆ NULL

```
<?php
$bool = true;    // Значение Boolean
$int = 100;     // Значение Integer
$string = "Переменная содержит текст"; // Значение String
$string2 = "5425"; // Значение String, так как число взято в кавычки !
$float = 44.122; // Значение Float
?>
```

```
<?php
$str = "50000"; // Значение String
$new_str = (integer) $str; // Теперь значение стало Integer

// Проверяем...

echo $new_str + $new_str;
?>
```



Операторы

Математические операторы и функции

- ❖ + сумма двух чисел
- ❖ - разность чисел
- ❖ / частное от деления двух чисел
- ❖ % остаток от деления

```
<?php
echo "2 + 2 = ", 2 + 2, "<br>";
echo "5 - 2 = ", 5 - 2, "<br>";
echo "10 * 10 = ", 10 * 10, "<br>";
echo "100 / 2 = ", 100 / 2, "<br>";
echo "10 % 2 = ", 10 % 2, "<br>";
?>
```

```
<?php
echo "round(4.2) = ", round(4.2), "<br>";
?>
```



Операторы присвоения

```
<?php  
$fruits = 14;  
?>
```

```
<?php  
$n = $m = $p = 3;  
echo $n, $m, $p;  
?>
```

Также в PHP есть комбинированные операторы, которые делают код более компактным. Вот их перечень:

+=, -=, /=, .=, %=, &=, |=, ^=, <=, >=



Инкремент и декремент

```
<?php
$a = $b = $c = $d = 2;

echo $a++, "<br>";
echo ++$b, "<br>";
echo $c--, "<br>";
echo --$d, "<br>";
?>
```



Оператор исполнения

В PHP существует такой оператор, как **оператор исполнения**, он нужен для того чтобы выполнять команды ОС и использовать результат этого выполнения.

Любая строка, которая заключена в обратные апострофы — ` ` считается как команда ОС. Например (как результат вы получите список директорий диска D):

```
<?php
$d = `dir d:\\`;
echo $d;
?>
```



Строковые операторы

PHP имеет два строковых оператора.

- ❖ **Первый** – оператор конкатенации `.`, который объединяет две строки в одну.
- ❖ **Второй** - конкатенирующий оператор присвоения `.=`, добавляет к строке нужное значение.

```
<?php
$d = "Hello";
$f = $d." world";    // Теперь $f = "Hello world"

echo $f;

echo "<br/>";

$f .= " !!!";    // Теперь $f = "Hello world !!!"

echo $f;
?>
```



Условные операторы

Условный оператор IF

```
<?php
$speed = 80;

if ($speed > 60)
{
    // Начало
    echo "Превышение скорости! <br>";
    echo "Пожалуйста, уменьшите скорость!";
}
    // Конец
?>
```



Операторы сравнения

Оператор	Операция
==	Равенство
===	Идентичность
!=	Неравенство
<>	Неравенство
!==	Неидентичность
>	Больше
>=	Больше или равно
<	Меньше
<=	Меньше или равно

```
<?php
$speed = 45;

if ($speed <= 60)
    echo "Скорость в пределах нормы";
?>
```



Логические операторы

Оператор	Операция
and	Логическое «И»
&&	Логическое «И»
or	Логическое «ИЛИ»
	Логическое «ИЛИ»
xor	Логическое «Исключающее ИЛИ»
!	Логическое «НЕ»

```
<?php
$speed = 40;

if ($speed > 35 && $speed < 55) {
    echo "Скорость в пределах нормы";
}
?>
```



Оператор ELSE

```
<?php
$speed = 50;

if ($speed > 60)
    echo "Превышение скорости !";
else
    echo "Скорость в пределах нормы"
?>
```



Оператор ELSEIF

```
<?php
$хspeed = 50;

if ($хspeed < 30)
    echo "Скорость в пределах нормы";

elseif ($хspeed >= 30 && $хspeed <= 60)
    echo "Ваша скорость {$хspeed} км/час";

else
    echo "Превышение скорости !";
?>
```

elseif



else if



Оператор SWITCH

```
<?php
$хspeed = 55;

switch($speed)
{
    case 30 :
        echo "Ваша скорость 30 км/час";
        break;

    case 58 :
        echo "Ваша скорость 50 км/час";
        break;

    case 70 :
        echo "Превышение скорости !";
        break;

    default :
        echo "Скорость в пределах нормы";
        break;
}
?>
```



Циклы

Цикл FOR

```
<?php
for ($i = 0; $i < 10; $i++)
{
    echo "Вывод строки. 10 раз <br>";
}
?>
```



Цикл WHILE

```
<?php
$counter = 0;
while ($counter < 5)
{
    echo "Эта строка выведется 5 раз <br>";
    $counter++;
}
?>
```



Цикл DO...WHILE

```
<?php
$counter = 6;

do
{
    echo "Эта строка выведется 1 раз <br>";
    $counter++;
}
while ($counter < 5);

?>
```



Цикл FOREACH

```
<?php
$array = array ("Apple", "Limon", "Chery", "Oranges");

foreach ($array as $value)
{
    echo "Вы выбрали фрукт - $value <br>";
}
?>
```



Строки

Функции для обработки строк

С помощью этих функций можно, например, обрезать строку, дописывать строку, заменить часть строки и много другое.

Пример:

```
<?php
echo substr("Hello world", 6, 5); // получим world
// Параметры:
// 1 - исх. строка
// 2 - с какого символа
// 3 - сколько символов (если не указано - до конца)

echo strpos("Hello world", "world"); // получим 6
// Параметры:
// 1 - исх. строка
// 2 - искомая подстрока
?>
```



Массивы

Создание массивов

Массивы создаются при помощи оператора присвоения, также как и переменная.

Имена массивов начинаются со знака \$, после которого следует произвольный идентификатор, далее идут квадратные скобки: `$arr[0] = "php";`

```
<?php
$arr[0] = "php";
$arr[1] = "html";
$arr[2] = "css";
?>
```

```
<?php
$arr["Kiev"] = 3000000;
$arr["Paris"] = 5000000;
$arr["LA"] = 15000000;
?>
```

```
<?php
$arr[] = 3000000;
$arr[] = 5000000;
$arr[] = 15000000;
?>
```



Для создания массива

```
<?php  
$arr = array("php", "html", "css");  
?>
```

```
<?php  
$arr = array(1 => "php", "html", "css");  
?>
```

```
<?php  
$arr = array("first" => "php", "second" => "html", "third" => "css");  
?>
```



Для создания массива

```
<?php  
$arr[0] = "PHP";  
$arr[1] = "HTML";  
$arr[2] = "CSS";  
?>
```

```
<?php  
$arr[1] = "JAVASCRIPT";  
?>
```

```
<?php  
$arr[] = "JQUERY";  
?>
```



Удаление элементов массива

Если нам нужно удалить один из элементов массива, то для этого мы должны использовать функцию **unset**

```
<?php
$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

unset($arr[1]);

foreach($arr as $key => $value) {
    echo $value.'<br/>';
}
?>
```



Перебор элементов массива

Для вывода всех элементов массива на экран можно использовать функцию `print_r`, которая выведет все элементы массива вместе с их индексами.

```
<?php
$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

print_r($arr);

// Отображение в браузере:
// Array ( [0] => PHP [1] => HTML [2] => CSS )
?>
```

```
<?php
$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

foreach($arr as $value)
{
    echo $value, "<br>";
}

// Отображение в браузере:
// PHP
// HTML
// CSS
?>
```



Функции для работы с массивами

Для работы с массивами в PHP предусмотрено очень много функций, полный список которых находится на странице официальной документации:

<https://www.php.net/manual/ru/ref.array.php>



Сортировка массивов

```
<?php
$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

sort($arr);

print_r($arr);
// Отображение в браузере:
// Array ( [0] => CSS [1] => HTML [2] => PHP )
?>
```

```
<?php
$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

rsort($arr);

print_r($arr);
// Отображение в браузере:
// Array ( [0] => PHP [1] => HTML [2] => CSS )
?>
```



Сортировка массивов

```
<?php
$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

ksort($arr);

print_r($arr);
// Отображение в браузере:
// Array ( [0] => PHP [1] => HTML [2] => CSS )
?>
```

```
<?php
$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

ksort($arr);

print_r($arr);
// Отображение в браузере:
// Array ( [0] => PHP [1] => HTML [2] => CSS )
?>
```

```
<?php
$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

krsort($arr);

print_r($arr);
// Отображение в браузере:
// Array ( [2] => CSS [1] => HTML [0] => PHP )
?>
```



Преобразование строк в массивы и наоборот

implode - формирует строку из массива.

explode - формирует массив из строки.

```
<?php
$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

$string = implode(", ", $arr);
echo $string;
// Отображение в браузере:
// PHP, HTML, CSS
?>
```

```
<?php
$string = "PHP, HTML, CSS";
$arr = explode(", ", $string);
print_r($arr);
// Отображение в браузере:
// Array ( [0] => PHP [1] => HTML [2] => CSS )
?>
```



Многомерные массивы

```
<?php
$companies["Microsoft"][1] = "Programmer";
$companies["Microsoft"][2] = "PR";
$companies["Microsoft"][3] = "Office Manager";

$companies["Google"][1] = "IT";
$companies["Google"][2] = "Web-design";

$companies["Mozilla"][1] = "PR";
$companies["Mozilla"][2] = "C++ Programmer";

print_r($companies);
// Отображение в браузере:
// Array (
// [Microsoft] => Array ( [1] => Programmer [2] => PR [3] => Office Maneger )
// [Google] => Array ( [1] => IT [2] => Web-design )
// [Mozilla] => Array ( [1] => PR [2] => C++ Programmer )
// )
?>
```



Использование циклов в многомерных массивах

```
<?php
// Создание массива
$companies[0][] = "Programmer";
$companies[0][] = "PR";

$companies[1][] = "IT";
$companies[1][] = "Web-design";

$companies[2][] = "PR";
$companies[2][] = "C++ Programmer";

// Вывод массива на экран
for($i = 0; $i < count($companies); $i++) {
    for($j = 0; $j < count($companies[$i]); $j++) {
        echo $companies[$i][$j], "<br />";
    }
}
?>
```



Функции

Функции в PHP

```
<?php
function square($num)
{
    $square = $num * $num;
    echo $square;
}

square(7);

?>
```

```
<?php
function myfunc($x,$y) {
    $res = cos($x) + sin($y) + 2;
    return $res;
}

$x = 5;
$y = 7;

$z = myfunc($x,$y);

echo $z;

?>
```

```
<?php
function hello($text = "Привет") {
    echo $text;
}

hello('Добрый день');

echo "<br/>";

hello(); // Значение по-умолчанию "Привет"
?>
```



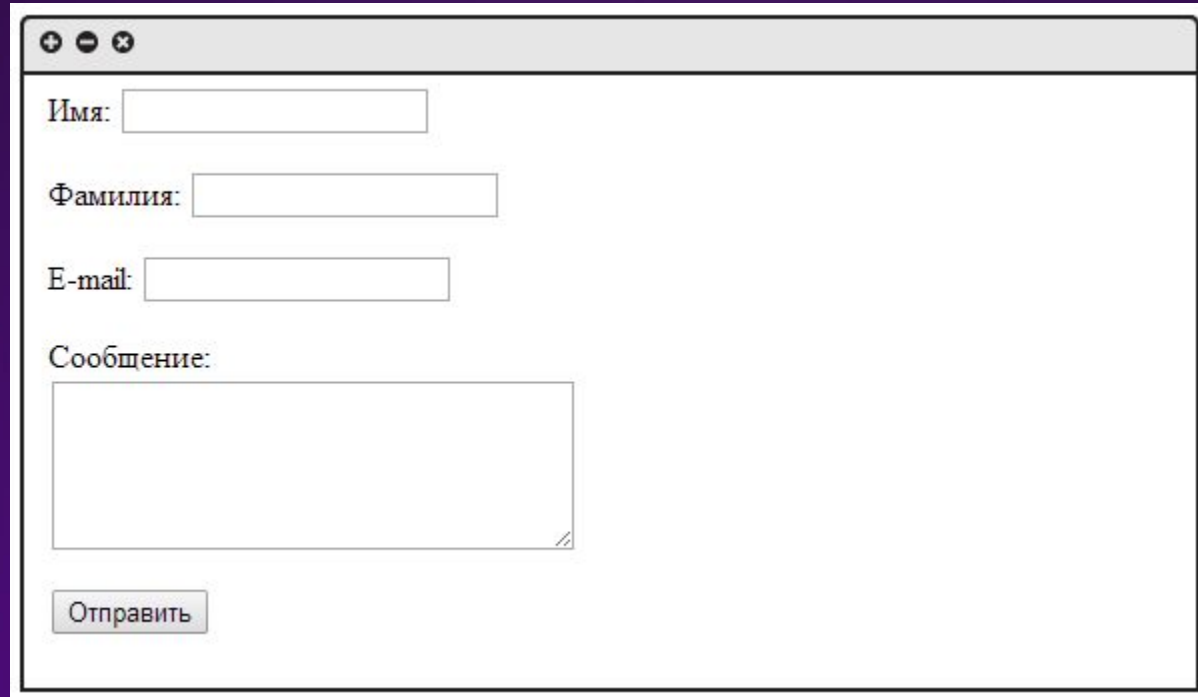
PHP + HTML

Получаем данные от элементов формы с помощью PHP

```
<form action="app/check.php" method="post">
<p>Имя: <input name="name" type="text"></p>
<p>Фамилия: <input name="surname" type="text"></p>
<p>E-mail: <input name="email" type="text"></p>
<p>Сообщение: <br /><textarea name="message" cols="30" rows="5"></textarea></p>
<p><input type='submit' value='Отправить'></p>
</form>
```



Получаем данные от элементов формы с помощью PHP



Имя:

Фамилия:

E-mail:

Сообщение:



Получаем данные от элементов формы с помощью PHP

1. Создайте и откройте (через редактор) обработчик `app/code.php` - сначала, это просто пустая страница. Далее откройте тег PHP - `<?php`.
2. Нужно проверить, была ли отправлена форма, для этого будем использовать глобальную переменную `$_SERVER` и проверять `REQUEST METHOD`

```
<?php
if($_SERVER['REQUEST_METHOD'] == 'POST') {
    // наш код
}
?>
```



Получаем данные от элементов формы с помощью PHP

3. Дальше, если форма отправлена, мы можем получить данные от поля "Имя", для этого, какой-нибудь переменной присваиваем полученное значение от этого поля, например:

```
<?php  
$name = $_POST['name'];  
?>
```



Получаем данные от элементов формы с помощью PHP

4. То же самое делаем и для остальных полей:

```
<?php
$name = $_POST['name'];

$surname = $_POST['surname'];
$email = $_POST['email'];
$message = $_POST['message'];
?>
```



Получаем данные от элементов формы с помощью PHP

5. Данные мы получили, теперь мы можем их вывести, для этого в страницу обработчика дописываем код:

```
<?php
$name = $_POST['name'];

$surname = $_POST['surname'];
$email = $_POST['email'];
$message = $_POST['message'];

echo $name."<br />".$surname."<br />".$email."<br />".$message."<br />";
?>
```



Проверка данных формы с помощью PHP

1. Давайте используем ту часть кода, где мы получили данные из формы:

```
<?php
$name = $_POST['name'];
$surname = $_POST['surname'];
$email = $_POST['email'];
$message = $_POST['message'];
?>
```



Проверка данных формы с помощью PHP

2. Теперь нам нужно проверить переданные нам данные. Чтобы не писать один и тот же код, давайте создадим несколько функций для проверки.

Сначала создадим функцию для очистки данных от HTML и PHP тегов:

```
<?php
function clean($value = "") {
    $value = trim($value);
    $value = stripslashes($value);
    $value = strip_tags($value);
    $value = htmlspecialchars($value);

    return $value;
}
?>
```



Проверка данных формы с помощью PHP

Дальше, создадим функцию для проверки длины строки:

```
<?php
function check_length($value = "", $min, $max) {
    $result = (mb_strlen($value) < $min || mb_strlen($value) > $max);
    return !$result;
}
?>
```



Проверка данных формы с помощью PHP

3. Нам нужно "прогнать" переменные через эти функции:

```
<?php
$name = clean($name);
$surname = clean($surname);
$email = clean($email);
$message = clean($message);

if(!empty($name) && !empty($surname) && !empty($email) && !empty($message)) {
    // ...
}
?>
```



Проверка данных формы с помощью PHP

Если значения не пустые (проверили с помощью функции `empty`), то можно продолжать проверку дальше:

```
<?php
if(!empty($name) && !empty($surname) && !empty($email) && !empty($message)) {
    $email_validate = filter_var($email, FILTER_VALIDATE_EMAIL);

    if(check_length($name, 2, 25) && check_length($surname, 2, 50) && check_length($message
        , 2, 1000) && $email_validate) {
        // ...
    }
}
?>
```



Проверка данных формы с помощью PHP

4. Давайте добавим сообщение об успешности операции, если данные прошли все проверки.

```
<?php
if(!empty($name) && !empty($surname) && !empty($email) && !empty($message)) {
    $email_validate = filter_var($email, FILTER_VALIDATE_EMAIL);

    if(check_length($name, 2, 25) && check_length($surname, 2, 50) && check_length($message
        , 2, 1000) && $email_validate) {
        echo "Спасибо за сообщение";
    }
}
?>
```



Проверка данных формы с помощью PHP

5. В конце, нам нужно добавить сообщения для уведомления о том, что данные не прошли проверку.

```
<?php
if(!empty($name) && !empty($surname) && !empty($email) && !empty($message)) {
    $email_validate = filter_var($email, FILTER_VALIDATE_EMAIL);

    if(check_length($name, 2, 25) && check_length($surname, 2, 50) && check_length($message
    , 2, 1000) && $email_validate) {
        echo "Спасибо за сообщение";
    } else { // добавили сообщение
        echo "Введенные данные некорректны";
    }
} else { // добавили сообщение
    echo "Заполните пустые поля";
}
?>
```



Практическое задание

Задача 1.

Переменная `$lang` может принимать два значения: «ru» и «en». Если она имеет значение «ru», то в переменную `$arr` запишем массив дней недели на русском языке, а если имеет значение «en» – то на английском.

Варианты:

- 1 – Решить через switch-case.
- 2 – Решить через if, else, elseif.

Практическое задание

Задача 2.

Заполните массив 10 случайными числами от 1 до 50. Выведите его. Получившийся массив разделить на 2 новых – массив четных и нечетных.

Практическое задание

Задача 3.

Считайте введенную с клавиатуры строку. Если в ней более 5 символов – возьмите 5 первых, дополните многоточием и выведите. Если 5 и менее символов – просто выведите строку.

Практическое задание

Задача 4. Вариант 1.

Считайте из формы имя, фамилию и возраст пользователя, запретите ввод тегов и обрежьте концевые пробелы. При правильном вводе после отправки скройте форму.

Практическое задание

Задача 4. Вариант 2.

Считайте из формы имя, фамилию и возраст пользователя, запретите ввод тегов и обрежьте концевые пробелы. При правильном вводе после отправки значения должны остаться в полях формы.

Оценивание

- ◆ 25 – 34 – задачи 1, 2.
 - ◆ 35 – 44 – задачи 1, 2, 3.
 - ◆ 45 – 50 – все задачи.
 - ◆ 51 – 54 – все задачи за выходные (до 27.10.2019).
-
- ◆ Смотрите заметки!
 - ◆ Тема письма: Группа | Семинар 5 | Фамилия Имя